REMARKS

Claims 1-17 are pending in the application. Claims 1-17 have been rejected.

Claims 1 - 8 and 11 - 15 stand rejected under 35 U.S.C. § 112 second paragraph. Claims 1 and 11 have been amended to address this rejection.

Claims 1 - 8 and 11 - 15 stand rejected under 35 U.S.C. § 102(b) as being anticipated by Tirumalai et al., U.S. Patent No. 6,341,370 B1 (Tirumalai).

Claims 9-10, 16 and 17 stand rejected under 35 U.S.C. § 102(e) as being anticipated by Chen, U.S. Patent No. 6,625,807 B1 (Chen).

Claims 1 – 8 and 11 – 15 are allowable over Tirumala.

The present invention, as set forth by independent claim 1 relates to a method of determining an execution order for machine instructions to reduce spill code, the method comprising the steps of scheduling the machine instruction for which an amount by which a size of a committed set of machine instructions would increase upon the scheduling of the machine instruction is smallest from machine instructions that are ready for scheduling, and determining an execution order for the machine instructions to reduce spill code.

The present invention, as set forth by independent claim 11 relates to a computer program product having media including computer programmed instructions for determining an execution order for machine instructions to reduce spill code, the computer program product including first subprocesses for scheduling the machine instruction for which an amount by which a size of a committed set of machine instructions would increase upon the scheduling of the machine instruction is smallest from machine instructions that are ready for scheduling, and determining an execution order for the machine instructions to reduce spill code.

Tirumala relates to executing instructions in a computer and more specifically to improving performance of assembly code generated by an optimizing compiler by integrating data prefetching and modulo scheduling and by inserting prefetch instructions generated after modulo scheduling has been performed. The insertion of prefetch instructions is performed in a manner which minimizes spills and reloads. (See e.g., Tirumula, Col. 4, lines 63 – 67.) When discussing avoiding spills, Tirumala sets forth:

> Avoiding spills and reloads requires two items. First, problem situations must be recognized. This is done by estimating the register pressure after scheduling, as noted above with regard to step 240. Second, mechanisms must be provided to take suitable action(s) after a problem situation has been recognized. If the bounds exceed the thresholds specified for each data type (i.e., the estimated register pressure is unacceptable), then the schedule should be discarded and a new one derived. These thresholds are usually just one or two less than the number of physical registers available. Scheduling parameters are then adjusted to reduce the register pressure to acceptable levels (step 250) and the loop is rescheduled. If the estimated register pressure is acceptable, the process proceeds to step 255, at which point prefetch calculations are performed and the KUF adjusted. This step is described in greater detail with regard to FIG. 3 (Tirumala, Col. 7, lines 47 – 65).

However, while Tirumala discusses avoiding spills, Tirumala does not disclose or suggest scheduling a machine instruction for which an amount by which a size of a committed set of machine instructions would increase upon the scheduling of the machine instruction is smallest, as required by claim 1 and as substantially required by claim 11.

More specifically, Tirumala does not disclose a method of determining an execution order for machine instructions to reduce spill code, the method comprising the steps of from machine instructions that are ready for scheduling, scheduling the machine instruction for which an amount by which a size of a committed set of machine instructions would increase upon the scheduling of the machine instruction is smallest, and determining an execution order for the machine instructions to reduce spill code, all as required by Claim 1 and as substantially required by claim 11. Claims 2-10 depend from Claim 1 and are allowable for this reason. Claims 12 – 15 depend from claim 11 and are allowable for at least this reason.

<u>Claims 9, 10, 16 and 17 are allowable over Chen.</u>

The present invention as set forth by Claim 9 relates to a method of determining an execution order for machine instructions to reduce spill code, the method comprising the steps of in a first bit vector containing one bit to represent each machine instruction to be scheduled, setting those bits for which the represented machine instruction is not committed, and resetting the remaining bits, for the each machine instruction to be scheduled that is ready for scheduling in a second bit vector also having one bit to represent each machine instruction to be ordered in the same sequence as in the first bit vector, setting those bits for which the represented machine instruction is a descendant of the each machine instruction that is ready for scheduling, and resetting the remaining bits, performing a bitwise AND operation of the first bit vector and the second bit vector to create a third bit vector, and determining the number of set bits in the third bit vector, and selecting for execution the machine instruction for which the third bit vector contains a minimum number of set bits.

The present invention as set forth by Claim 16 relates to a system in a computing environment for determining an execution order for machine instructions to reduce spill code, the system comprising first means for, in a first bit vector containing one bit to represent each machine instruction to be scheduled, setting those bits for which the represented machine instruction is not committed, and resetting the remaining bits, second means for, for the each machine instruction to be scheduled that is ready for scheduling in a second bit vector also having one bit to represent each machine instruction to be ordered in the same sequence as in the first bit vector, setting those bits for which the represented machine instruction is a descendant of the each machine instruction that is ready for scheduling, and resetting the remaining bits, performing a bitwise AND operation of the first bit vector and the second bit vector to create a third bit vector, and determining the number of set bits in the third bit vector; and third means for selecting for execution the machine instruction for which the third bit vector contains a minimum number of set bits.

Chen relates to register usage optimization and more particularly to obtaining and using register usage information for register optimization during software translation. Chen discloses a compiler to produce a bit vector for each program unit (i.e., subroutine, function, and/or procedure). Each bit in the bit vector represents a particular caller saved register. A bit is set if the compiler uses the corresponding register within that program unit. During the translation, the translator examines the bit vector to determine which registers are free. Chen sets forth that a goal of the register allocation is to keep as much data in the register as possible to minimize load/store instructions for the spilled data. (See e.g., Chen, Col. 5, lines 43 – 50.)

While Chen discloses bit vectors, Chen does not disclose or suggest a bit vector containing one bit to represent each machine instruction to be scheduled, setting those bits for which the represented machine instruction is not committed and resetting the remaining bits, as required by claim 9 and substantially required by claim 16. Additionally, Chen does not disclose or suggest a second bit vector also having one bit to represent each machine instruction to be ordered in the same sequence as in the first bit vector, setting those bits for which the represented machine instruction is descendant of each machine instruction that is ready for scheduling and resetting the remaining bits, as required by claim 9 and substantially required by claim 16.

Accordingly, Chen could not disclose or suggest performing a bitwise AND operation on the first bit vector and the second bit vector to create a third bit vector or determining a number of set bits in the third bit vector, much less selecting for execution the machine instruction for which the third bit vector contains a minimum number of set bits, as required by claim 9 and substantially required by claim 16. Thus, claims 9 and 16 are allowable over Chen. Claim 10 depends from claim 9 and is allowable for at least this reason. Claim 17 depends from claim 16 and is allowable for at least this reason.

CONCLUSION

In view of the amendments and remarks set forth herein, the application is believed to be in condition for allowance and a notice to that effect is solicited. Nonetheless, should any issues remain that might be subject to resolution through a telephonic interview, the examiner is requested to telephone the undersigned.

The Commissioner is hereby authorized to charge Deposit Account No. 090461for any fees due and credit any overpayments to same.

I hereby certify that this correspondence is being electronically submitted to the COMMISSIONER FOR PATENTS via EFS on October 22, 2007.

/Stephen A. Terrile/

Attorney for Applicant(s)

Respectfully submitted,

/Stephen A. Terrile/

Stephen A. Terrile
Attorney for Applicant(s)
Reg. No. 32,946